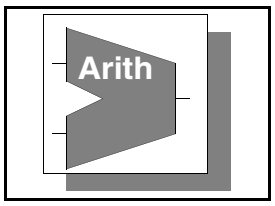
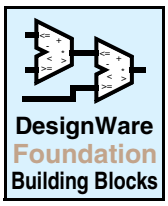




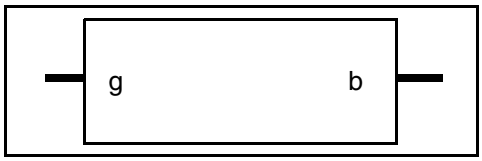
DW_gray2bin

Gray to Binary Converter

Version, STAR, and myDesignWare Subscriptions: IP Directory

Features and Benefits

- Parameterized word length
- Inferable using a function call



Description

DW_gray2bin converts Gray coded input `g` to binary coded output `b`.

Table 1-1 Pin Description

Pin Name	Width	Direction	Function
g	*width* bits	Input	Gray coded input data
b	*width* bits	Output	Binary coded output data

Table 1-2 Parameter Description

Parameter	Values	Description
width	≥ 1	Input word length

Table 1-3 Synthesis Implementations

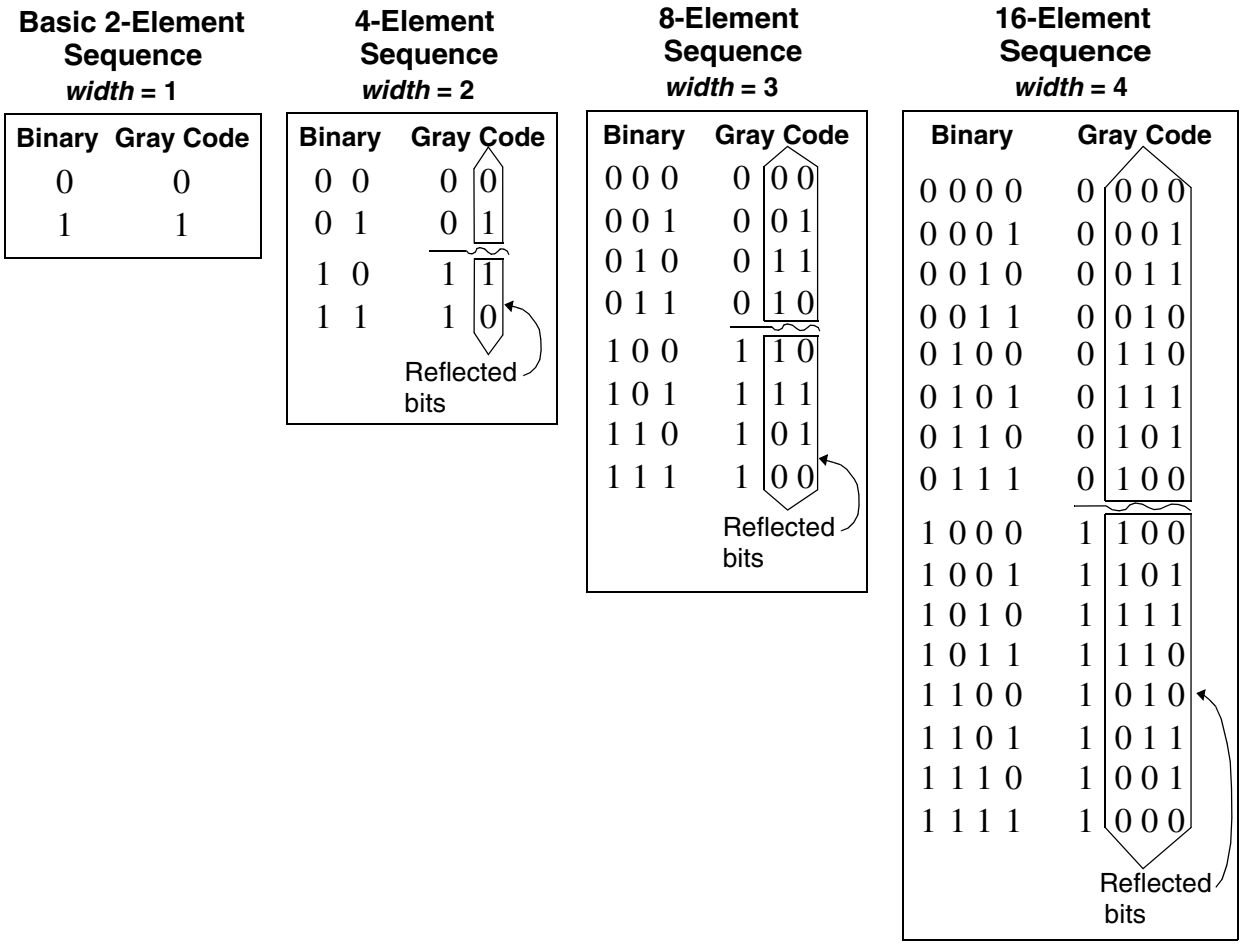
Implementation Name	Function	License Feature Required
rpl	Ripple-carry synthesis model	DesignWare
cla	Carry-lookahead synthesis model	DesignWare

Table 1-4 Simulation Models

Model	Function
DW01.DW_gray2bin_cfg_sim	Design unit name for VHDL simulation
dw/dw01/src/DW_gray2bin_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW_gray2bin.v	Verilog simulation model source code

Reflected binary Gray code sequences can be constructed iteratively starting with the simplest two element sequence of 0 and 1 (see Figure 1-1). Each iteration doubles the sequence by concatenating the previous sequence with a reversed (reflected) copy of itself. In addition, a new Most Significant Bit (MSB) is added to each element with its value being 0 for the forward copy and 1 for the reflected copy.

Figure 1-1 Gray Code Number Relationship to Corresponding Binary Number



Related Topics

- Math – Arithmetic Overview
- DesignWare Building Block IP User Guide

HDL Usage Through Function Inferencing - VHDL

```
library IEEE, DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DW_Foundation_arith.all;

entity DW_gray2bin_func is
  generic (func_width : positive := 8);
  port (func_g : in  std_logic_vector(func_width-1 downto 0);
        b_func : out std_logic_vector(func_width-1 downto 0));
end DW_gray2bin_func;

architecture func of DW_gray2bin_func is
begin
  -- function inference of DW_gray2bin
  b_func <= DWF_gray2bin (func_g);
end func;
```

HDL Usage Through Function Inferencing - Verilog

```
module DW_gray2bin_func (func_g, b_func);

  parameter func_width = 8;

  input  [func_width-1 : 0] func_g;
  output [func_width-1 : 0] b_func;

  // pass "width" parameters to the inference functions
  parameter width = func_width;

  // Please add search_path = search_path + {synopsys_root + "/dw/sim_ver"}
  // to your .synopsys_dc.setup file (for synthesis) and add
  // +incdir+$SYNOPSYS/dw/sim_ver+ to your verilog simulator command line
  // (for simulation).
  `include "DW_gray2bin_function.inc"

  // function inference of DW_gray2bin
  assign b_func = DWF_gray2bin (func_g);
endmodule
```

HDL Usage Through Component Instantiation - VHDL

```
library IEEE, DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DW_Foundation_comp_arith.all;

entity DW_gray2bin_inst is
  generic (inst_width : positive := 8);
  port (inst_g : in  std_logic_vector(inst_width-1 downto 0);
        b_inst : out std_logic_vector(inst_width-1 downto 0));
end DW_gray2bin_inst;

architecture inst of DW_gray2bin_inst is
begin
  -- instance of DW_gray2bin
  U1 : DW_gray2bin
    generic map (width => inst_width)
    port map (g => inst_g,
              b => b_inst);
end inst;

-- pragma translate_off
configuration DW_gray2bin_inst_cfg_inst of DW_gray2bin_inst is
  for inst
  end for;
end DW_gray2bin_inst_cfg_inst;
-- pragma translate_on
```

HDL Usage Through Component Instantiation - Verilog

```
module DW_gray2bin_inst (inst_g, b_inst);

  parameter inst_width = 8;

  input  [inst_width-1 : 0] inst_g;
  output [inst_width-1 : 0] b_inst;
  // Please add +incdir+$SYNOPSYS/dw/sim_ver+ to your verilog simulator
  // command line (for simulation).
  // instance of DW_gray2bin
  DW_gray2bin #(inst_width)
    U1 (.g(inst_g),
        .b(b_inst));
endmodule
```

Copyright Notice and Proprietary Information

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.

All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
www.synopsys.com